



15/02/21

CIRCUITI INTEGRATI

Composti da transistor, microscopici interruttori.
Servono a realizzare macchine digitali.

MACHINE DIGITALI

È un sist. digitale per immagazzinare, elaborare e comunicare info, impiegando segnali digitali \rightarrow è insieme finito/discreto/digitale di valori significativi.

SEGNALI DIGITALI E ANALOGICI

• ANALOGICI \exists nella realtà

Variano in modo continuo all'interno di insieme di valori ammissibili.
L'info è data da ogni possibile valore. Un disturbo del segnale distrugge l'info (rumore) e rende il tutto poco robusto e per ottenerne queste info servono strumenti sofisticati.

• DIGITALI \exists ma li abbiamo creati per comodità

È l'intervalllo a determinare l'info, rappresenta meno info, ha più robustezza, minor complessità e costo.

Avviene digitalizzando i valori di ingresso.

Per avere la massima robustezza si possono individuare 2 intervalli che identificano un SEGNALE BINARIO.

CODICI BINARI

Si cerca il più possibile di tenerli verso gli estremi, per evitare incertezze intorno al limite tra i 2 intervalli.

H \rightarrow 0 \rightarrow logica negativa

L \rightarrow 1 \rightarrow logica positiva

Ma con un unico bit posso solo indicare 2 cose, quindi uso più bit e vado a formare una stringa che identifica un altro elemento. E con n bit posso ottenere 2^n stringhe.

Il bit più a sx è il BIT PIÙ SIGNIFICATIVO, viceversa per quello più a dx.

Il codice è la corrispondenza arbitraria tra info e configurazione, condivisa tra sorgente e destinazione.

INTERRUTTORE

COMMUTA⁺

Ha 2 posizioni e viene chiuso o aperto e cambia il suo stato in base alle esigenze.

ASTRAZIONE

Creiamo degli elementi che rappresentano ciò che fanno i transistor ad esempio e utilizziamo questi strumenti teorici per gestire la complessità.

22/02/21

RETE LOGICA

È un'astrazione della combinazione di interruttori. Ciò viene fatto per gestire la complessità e a tal fine si definiscono dei componenti ELEMENTARI, i GATE

GATE 1 INGRESSO

Il numero di f. è di n ingressi binari con un'uscita binaria c'è: 2^n

- GATE NOT

TAB.

X	Y
0	1
1	0

SIMB.



EXPR.

$$\begin{aligned}Z &= \overline{X} \\ Z &= X' \\ Z &= X^* \\ Z &= !X\end{aligned}$$

GATE 2 INGRESSI

- GATE AND \rightarrow ASTR. AI CONTATTI IN SERIE

TAB.

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

SIMB.

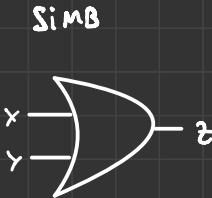


EXPR.

$$\begin{aligned}Z &= X \cdot Y \\ Z &= XY\end{aligned}$$

• GATE OR → ASTR. DI CONTATTI IN PARALLELO

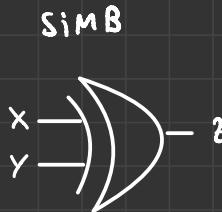
TAB	x	y	z
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1



EXPR.
 $z = x + y$

• GATE EXOR → ASTR. DI 2 DEVIATORI

TAB	x	y	z
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0



EXPR
 $z = x \oplus y$

• Viene anche detto SOMMA MODULO 2

• Si può interpretare come L'USCITA VALORE 1 \Leftrightarrow IL NUM. DI 1 IN INGRESSO
 È PARI
 Lo utile x FAN-IN > 2

• GATE NEGATI

Si distinguono dal "o" alla fine nel simbolo

I gate NAND, NOR, EXNOR

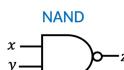
Tabella della Verità

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1

Simbolo



Espressione

$$z = x \uparrow y$$

oppure

$$z = \overline{x}y$$



$$z = x \downarrow y$$

oppure

$$z = \overline{x} + \overline{y}$$



$$z = x \equiv y$$

oppure

$$z = x \oplus y$$

→ ANCHE DETTO EQUIVALENCE

si potrebbero
 ottenere
 applicando
 un GATE NOT
 dopo i rispett.
 GATE non
 negati

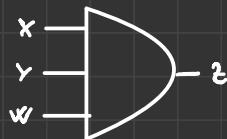
GATE CON INGRESSI > 2 → FAN-IN > 2

es AND con FAN-IN = 3

TAB

x	y	w	z
0	0	0	0
0	0	1	0
0	1	1	0
1	1	1	1
1	0	1	0
1	0	0	0
1	1	0	0
0	1	0	0

SIMB.



EXPR.

$$z = x \cdot y \cdot w$$

DIAGRAMMI AD OCCITÒ

Vengono usati per rappresentare l'evoluzione di gruppi di segnali.

Le linee hanno un valore.



rappr. di un GATE AND

I segnali non cambiano istantaneamente quindi ci sono dei momenti transizioni

BUS DI SEGNALI

Sono un insieme di segnali che possono essere descritti con una config. binaria nel diagramma.

Le linee non hanno più valore, che invece è espresso dai bit all'interno.

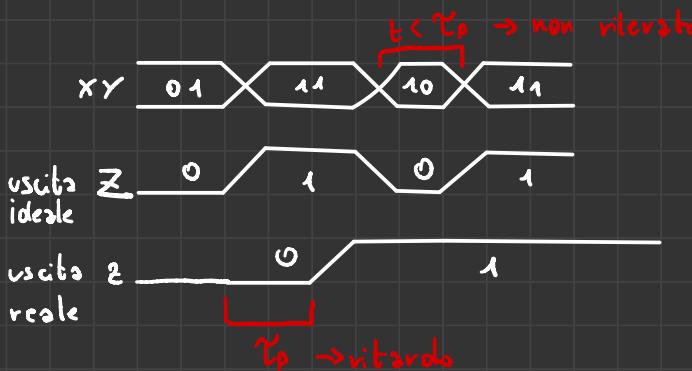


rappr. di GATE AND

RITARDI DI PROPAGAZIONE

La propagazione del segnale non è immediata e quindi l'uscita non cambia subito ma impiega un tempo τ_p .

Si tratta di un ritardo inerziale che stabilisce un limite superiore per la velocità di funzionamento



rappr di GATE
AND ideale e reale

POSSIBILI MONTAGGI

- SERIE
l'uscita di un gate è l'ingresso di un'altro
- PARALLELO
alcuni ingressi sono in comune
- RETROAZIONE
come in serie ma viceversa

26/02/21

INFORMAZIONE

- STRINGA DI LUNGHEZZA FINITA DI SIMBOLI E ALFABETO → TEORIA INFORMAZIONE
Noi impieghiamo un alfabeto binario composto da 0 e 1 e l'info è una stringa di bit

es. CODIFICA

$$2^4 > 10 \rightarrow 4 \text{ bit}$$

con più bit sì con meno bit no

CODICE BINARIO

Funzione dall'insieme delle 2^n configurazioni di n bit ad un insieme delle M informazioni e ci deve essere n tale che $2^n \geq M$

PROPRIETÀ CODICE

- scelta codifica
- num. bit
- associazione
- codici possibili $C = \frac{2^n!}{(2^n - M)!}$

CODICE RIDONDANTE

Codice che usa il numero non minimo di bit

numero minimo : $n_{\min} = \log_2(M)$

- oss: In un codice non ridondante possono essere config. non usate
Il grafico di codifici (p.62) ha una crescita logaritmica standardizzata
es. cod. non ridondante → codifica BCD codifica le prime 10 cifre decimali
es. cod. ridondante → "1 su n" ha un uno nel pos corrisp. alla cifra
"7 segmenti" si fa accendersi la parte della cifra con
uno 0 e si codificano i numeri gli
conseguenze
- I codici ridondanti servono quando bisogna interagire con umani e rendere
semplice l'informazione da leggere
"UPC" è usato nel codice a barre,

RAPPRESENTAZIONE POSIZIONALE

Il concetto di numero è astratto ed è indipendente dalla base

- Un numero N in base $\beta \geq 2$ è rapp. da $n+m$ cifre c_j
 $(N)_\beta = (c_{n-1}, \dots, c_0, c_{-1}, \dots, c_{-m})_\beta$

Per basi < 10 si usano cifre altrimenti caratteri

Ognuna delle β cifre corrisponde a un \neq num naturale $\in [0, \beta-1]$
 $c_k \rightarrow n_k \in \{0, 1, \dots, \beta-1\} \forall k$

- Per calcolare il valore di N si fa

$$N = (n_{n-1} \cdot \beta^{n-1} + \dots + n_0 \cdot \beta^0 + n_{-1} \cdot \beta^{-1} + \dots + n_{-m} \cdot \beta^{-m})$$

- STUDIEREMO

decimale $\beta = 10$ $[0, 9]$ $(1001)_{10} = 1001$

binario $\beta = 2$ $[0, 1]$ $(1001)_2 = 9$

esadecimale $\beta = 16$ $[0, 9] \cup [A, F]$ $(1001)_{16} = 4097$

CAMBIO $\beta=2 \leftrightarrow \beta=16$

Rappresentare una lunga stringa da $\beta=2$ a $\beta=16$ la compatta e ne aumenta i caratteri e quindi rende più chiaro

es. $(101011110001)_2$ = $(AF1)_{16}$

CAMBIO BASE IN GENERALE

A noi interessa $\beta: 10 \rightarrow 2 \circ 16$

$$2 \circ 16 \rightarrow 10$$

- CONVERSIONE ITERATIVA (parte intera, parte frazionaria)

p. intera: divisione per β con resto e cifra = resto

p. frazionaria: moltiplicazione per $\beta = \text{num}(\text{num. int}, \text{num. fraz.})$ poi
cifra = num. int e ricomincio con num. fraz.

non sempre finisce, a volte bisogna stabilirlo prima

NUMERI BINARI SENZA SEGNO

Nelle macchine digitali si usa base 2 , abbiamo numero di cifre finito a disposizione

→ con n bit otteniamo i numeri compresi fra 0 e $2^n - 1$

1/03/24

7³

CODICE ASCII (ug67)

A 7 bit, va interpretato come matrice con in alto i bit significativi e da lato la parte delle strings di bit che vanno a rappresentare il carattere
4 bit

All'interno abbiamo scelte di efficienza come maiuscole minuscole e del carattere DEL (ignora)

CODICE ASCII ESTESO

Diventa a 8 bit e nei diversi paesi quel bit in più viene gestito in modo x, ad esempio in occidente vengono aggiunte le lettere accentate

Poi venne esteso a 16 bit, ma ancora non bastavano per tutte le lingue con ideogrammi

Venne ancora esteso a 24 bit ma non doveva essere raggruppato in byte (1 byte = 8 bit) e viene fatto l'ENCODING ovvero vengono mappati i caratteri in sequenze di byte non ridondanti.

Vengono introdotti gli UTF:

UTF-32 → aggiungo 11 0 per avere multipli di 4 → spreco

UTF-16 → usa 2 byte per i caratteri più comuni e 4 per gli altri

UTF-8 → retrocompatibile con Ascii originario con 1 byte

con lingue con 2 byte

con altri caratteri con 3-4 byte

UTF-8, UTF-16 sono codici a lunghezza variabile in base al tipo di info e quindi sono molto efficienti

In utf-8 i codici

$\leq (7F)_{16} \rightarrow 1 \text{ byte}$

$\leq (7FF)_{16} \text{ e } > (7F)_{16} \rightarrow 2 \text{ byte}$ primo con 110, il secondo con 10

$\leq (FFFF)_{16} \text{ e } > (7FF)_{16} \rightarrow 3 \text{ byte}$ primo con 1110, gli altri con 10

BIT MAP

Rappresentazione ridondante, un font, che usano molti bit di cui solo alcuni vengono scelti per rappresentare il carattere.

RETE LOGICA

Modello astratto che permette di analizzare e sintetizzare delle strutture.

Adesso gli ingressi hanno dei nomi perché non sarà più la stessa cosa, non sono commutabili perfetta.

es. trascodifica BCD / 7 segmenti



MACCHINA COMBINATORIA

Esegue comandi, non ha memoria, vive nel presente solo ingressi curr.

MACCHINA NON COMBINATORIA → SEQUENZIALE

Dove memorizzare una sequenza e in base a tale ordine cambia l'output.

es. cassaforte, CPU, semaforo

COMB. vs SEQ.

Una macchina è seq. se a fronte degli stessi ingressi deve produrre un'uscita ≠ all'altro. È combinatoria.

MACCHINE COMBINATORIE

Hanno n ingressi e m uscite, che si possono intendere come m risultati di n incognite.

Le uscite possono essere analizzate singolarmente e considerare la rete finale come un insieme di reti ad una uscita.

La struttura (come è realizzata la rete) si può rappresentare con espressioni e schemi logici.

Il comportamento (cosa fa) si può rappr. con linguaggio naturale e la tab. della verità.



05/03/21

FUNZIONI COMPLETE E INCOMPLETE

② Per ogni 2^n config. di n ingressi è definito il val. di uscita z .
es. decoder, sommatore

① C'è almeno una delle 2^n config. di n ingressi per cui non è specificato il val. di uscita. (~~è oppure non interessa~~)
es. encoder, convertitore BCD/7 seg.

RETE LOGICA COMBINATORIA

Si può rappresentare come sistemi di n reti ad 1 uscita e ogni rete è una funzione a valori binari

TABELLE VERITA' FUNZ. INCOMPLETA

Ha 2^n righe e $n+1$ colonne, n per le incognite e 1 per l'uscita.

Nell'uscita, oltre a 0 e 1, Noi possiamo mettere un "-" che indica che quel valore non va specificato e non ci interessa il valore.

CONVERTITORE BCD/7 SEG.

Ci sono 7 funz. in // con 4 ingressi e 1 uscita.

SINTESI

Come si va dal comportamento alla struttura?

ALGEBRA BINARIA

Sist. matematico formato da operazioni che lavorano su variabili binarie e definiscono funzioni.

Intuizione data da Claude Shannon che riprese e formalizzò Boole.
Ne esistono diverse

ALGEBRA DI COMMUTAZIONE

La usa Shannon

C'è la somma, il prodotto e la complementazione (vedi p. 21)
rLOGICA X ARITMETICA

Corrispondono al comportamento di OR, AND, NOR.

SI MESI

A una TDV si associano vari SL, per individuare quale ci sono vari modi, uno dei più semplici è attraverso le espressioni canoniche

ESPRESSIONI CANONICHE

• SP - somma prodotti

Ogni funzione di n variabili ha una somma di tanti prodotti logici quante sono le config. per cui vale 1.

Ciasun prodotto si chiama MINTERMINE e ha una variabile vera se 1 e negata se 0 → voglio che tutti valgano 1

Quindi si GUARDA LA TDV E PRENDO LE RIGHE IN CUI L'USCITA E' 1, Poi VENGONO FATTI IOTI PRODOTTI CON QUELLE VARIABILI CHE VENGONO CAMBIATI.

SE VALGONO 1 SI LASCIANO COSÌ, ALTRIMENTI SI NEGANO E Poi si SOMMANO i PRODOTTI

• PS - prodotto di somme

Ogni funzione di n variabili ha un prodotto di tante somme logiche quante le config. per cui vale 0.

Ciasuna somma si chiama MAXTERMINE e ha una variabile vera se 0 e negata se 1.

PROCEDIMENTO QUASI IDENTICO A SP

⚠️ Quando nelle espressioni c'è lo stesso gate, allora si disegna solo 1 volta!

NOTAZIONE COMPATTA x espressioni canoniche

m - mintermine

M - maxtermine

EQUIVALENZA TRA ESPRESSIONI

Due espressioni sono equivalenti se rappresentano la stessa TSV ma possono avere complessità diverse.

PROPRIETÀ DI SOMMA e PRODOTTO

• commutatività $\rightarrow x+y = y+x \quad e \quad x \cdot y = y \cdot x$

• associatività $\rightarrow (x+y)+z = x+y+z \quad e \quad (x \cdot y) \cdot z = x \cdot y \cdot z$

• utile per scomporre le entrate

• distributività $\rightarrow (x \cdot y) + (x \cdot z) = x \cdot (y+z)$
e

$(x+y) \cdot (x+z) = x+(y \cdot z)$ non valida in algebra normale!

• idempotenza $\rightarrow x+x = x \quad e \quad x \cdot x = x$

• identità $\rightarrow x+0 = x \quad e \quad x \cdot 1 = x$

• limite $\rightarrow \underline{x+1=1} \quad e \quad x \cdot 0 = 0$

ATTENZIONE

• involuzione $\rightarrow (x')' = x$

• limitazione $\rightarrow x+(x')'=1 \quad e \quad x \cdot (x')'=0$

• combinazione $\rightarrow x \cdot y + x \cdot y' = x \quad e \quad (x+y) \cdot (x+y') = x$

$\hookrightarrow x(y+y') \rightarrow x \cdot 1 \rightarrow x$

• leggi di DeMorgan:

1) $(x+y)' = x' \cdot y'$

2) $(x \cdot y)' = x' + y'$

utili per cambiare i gate

→ Per passare da OR a AND e viceversa dovo o negare l'uscita o tutti gli ingressi

utili per le semplificazioni

8/03/21

EQUIVALENZE NOTEVOLI

• consenso $xy + x'z + yz = xy + x'z \quad e \quad (x+y) \cdot (x'+z) \cdot (y+z) = (x+y) \cdot (x'+z)$

RITARDI E VELOCITÀ

Cosa vuol dire rete veloce?

Ogni gate ha un suo ritardo τ_g e il ritardo complessivo è dato dalla somma dei ritardi sul percorso più lungo (si considera il ritardo peggiore).

I gate non hanno lo stesso ritardo, ma dobbiamo fare delle appross. e prendendo il caso peggiore è tutto più comodo.

COMPLESSITÀ E VELOCITÀ

• $N_{gate} \rightarrow$ numero gate + sono $\rightarrow +$ complesso

• $N_{conn} \rightarrow$ numero $\overset{\text{INGRESSI GATE}}{\text{connessioni}}$ + sono $\rightarrow +$ complesso

• $N_{casc} \rightarrow$ numero gate in cascata - sono $\rightarrow +$ velocità

RETI COSTO MINIMO

È lo schema logico che realizzano una funzione con:

- non più di 2 gate in cascata
- min. numero di gate
- min. numero di ingressi

Δ In generale N_{conn}, N_{gate} sono \neq tra PS e SP

Δ È possibile che più espressioni dello stesso tipo siano minime

ESPRESSIONE IRRIDONDANTE

Una espressione SP/PS che se la togliamo un termine non si ha più un'equivalenza con l'espressione stessa

↓
MODO ALGORITMICO PER VEDERE SE L'ESPRESSIONE E' IRRIDONDANTE

IMPLICANTI e IMPLICANTI PRIMI

• IMPLICANTI

Termine prodotto di n o meno variabili che assume 1 solo per config. in cui la funzione vale 1 o indifferenza.

3 INGRESSI : $a'b'c'$, $a'b'c$, $a'bc'$, $a'bc$, $ab'c'$, abc
2 INGRESSI : ab' , $a'b$, $a'c$, $a'c'$, $b'c'$, bc
1 INGRESSO : a'

per essere
implicanti,
deve valere in
ogni riga

• IMPLICANTI PRIMI

Implicante che cessa di essere tale rimuovendo un suo letterale
 a' , $b'c'$, bc

• IMPLICANTI PRIMI ESSENZIALI

Implicante che è l'unico ad assumere il valore 1 per alcune config. delle variabili di ingresso in cui la funzione assume valore 1.

a , bc

15/03/21

MAPPE DI KARNAUGH ~ CARNO ~

E' una tabella delle verità in un formato bidimensionale.

Ci sono molte permutazioni che hanno lo stesso significato.

Una regola è che i valori sulle righe e sulle colonne non devono differire per più di un bit alla volta. **CODICE DI GRAY**

Il motivo è che diventa più facile individuare gli implicanti.

Le celle sono adiacenti quando hanno un lato in comune, considerando anche "l'effetto pack-man"

Si può estendere anche alle mappe a 5/6 variabili con 1/2 bit per decidere in quale sottomappa mi trovo e la cella adiacente sarà la stessa di quella iniziale ma in un'altra sottomappa.

E nel caso delle 6 variabili per scegliere le mappe vediamo quelle adiacenti a quella originali.

- Se ho celle adiacenti allora guardo quali lettere sono presenti in entrambi per ottenere un implicante **x COMBINAZIONE**
se celle valgono 1

- Se ho una indiff. e un 1/0, mi viene dare 1/0 all'indiff per riandarvi al caso precedente

- Se ho h min/max termini adiacenti, posso usare la combinazione per 2 volte e ottenere un termine con 2 letterali in meno

RAGGRUPPAMENTO RETTANGOLARE (RR)

Insieme di 2^p celle dove ci sono le p colonne adiacenti di una mappa.

Se faccio raggruppamenti di 1,- trovo un IMPLICANTE

Se faccio raggruppamenti di 0,- trovo un IMPLICATO

Se questi riesco ad inserirli in gruppi maggiori, che li contengono interamente, allora sono degli IMPLICANTI / IMPLICATI PRIMI

Per individuare gli implicanti primi essenziali devo usare il consenso e posso individuare il fornire ridondante se il suo raggruppamento è individuato da altri raggruppamenti.

COPERTURA MINIMA

Una copertura è un insieme di RR che copre tutti gli 1 o 0 o -

La copertura minima è la copertura col minor numero di RR con dimensione maggiore possibile

NON C'È SEMPRE UN'UNICA COPERTURA MINIMA

22/03/21

INDIVIDUAZIONE TRAMICIA EPR. MINIMA

- 1) Decidere se fare PS o SP
- 2) si cerca il RR massimo, e si fa per tutte le possibili
- 3) si scelgono le coperture migliori
- 4) ripeti il 3 fino a soddisfare le cond. di copertura e si scrive expr. min.

NAND E NOR

Permettono di comporre circuiti usando meno gate.

Per fare la sintesi chi NAND parla da un SP.

Il NAND non è associativo!

Nella sintesi il num. di operatori è lo stesso

ALGORITMO SINTESI → p. 43

$$\text{es. } U = ab' + a'b = (a \cdot b') + (a' \cdot b) = (a \uparrow b') \uparrow (a' \uparrow b) = \\ = (a \uparrow (b \uparrow b')) \uparrow ((a \uparrow a) \uparrow b)$$

oppure

$$U = ab' + a'b = ab' + a'b + a'b + b'b = a(b' + b) + b(a' + b') = \\ = (a \cdot (b' + b)) + (b \cdot (a' + b')) = (a \uparrow (a \uparrow b)) \uparrow (b \uparrow (a \uparrow b))$$

Per fare la sintesi con i NOR parto da PS
Il NOR non e' associativo!

ALGORITMO sintesi → p.

26/03/24

DECODER $n : 2^n$

- Rete trascodifica da binario a "1 su 2^n "
- gli ingressi vengono anche chiamati indirizzi $[A_0, \dots, A_{n-1}]$ e A_0 è l'indirizzo di minor peso e A_{n-1} l'opposto
- l'indice i dell'uscita U_i ($0 \leq i \leq 2^{n-1}$) viene definito
 $i = A_{n-1} \cdot 2^{n-1} + \dots + A_0 \cdot 2^0$
- Realizza tutti i possibili mintermini di quelle variabili

FAN-OUT

Non è il numero di uscite!

È il numero di altri gate che posso collegare all'uscita del mio gate.
È un limite locale, di un solo gate. Non sono cumulativi.

FAN-IN

È il numero di ingressi di un gate, ci sono dei limiti fisici anche qui e quindi si può usare la prop. ass. Per AND, OR, EXOR oppure se ho un NAND o NOR posso applicare un NOT a volte

ENABLE

Segnale che vale 0 o 1 e che viene collegato con degli AND per avere il comportamento precedente con 1 o a portare tutto a 0 con 0 (DISABILITÀ USCITE).

NON LE SPAGNA PERÒ, C'È SEMPRE UN SEGNALE!!!

MULTIPLEXER

- E' un selettore a n vie, equivale a un "if"
- Decide il valore di uscita in base a uno tra gli n bit di ingresso che sceglie tra le $2^n - 1$ vie d'ingresso.
- Al crescere di n cresce esponenzialmente il num. vie

TEOREMA ESPANSIONE SHANNON

$$F(x_1, \dots, x_i, \dots, x_n) = x_i \cdot F(x_1, \dots, 1, \dots, x_n) + \bar{x}_i \cdot F(x_1, \dots, 0, \dots, x_n)$$

ESPRESSIONI GENERALI

Applicando ripetutamente shannon ottengo che

$$\text{SP: } F(x_1, \dots, x_i, \dots, x_n) = \sum_{i=0}^{2^n-1} m(i) \cdot F(i) \quad \left[\begin{array}{l} \text{TUTTO INSIEME DIVENTA UN} \\ \text{COMPONENTE PROGRAMMABILE} \\ \text{"LUT"} \end{array} \right]$$

LAVORO FATTO DAL MULTIPLEXER

MEMORIA

- Una memoria non modificabile e' a tutti gli effetti una TDR
- Le ROM (Read Only Memory) c'è un circuito che contiene ad ogni indirizzo un valore fissato.
- Le ROM sono reti programmabili, ogni uscita e' realizzata con una sintesi DECODE + OR (2^n AND + 1 OR). CONCETTUALMENTE la programmazione non avviene attraverso segnali esterni ma con contatti

29/03/21

ARITMETICA BINARIA SENZA SEGNO \nearrow CARRY OUT

- Quando faccio la somma e ho un riporto esterno di 1 allora il risultato è rapp. altrimenti no.

FULL - ADDER

fa restituire somma e riporto di due bit nel caso generale.
viene composto da più half-adder

HALF - ADDER

Non gestisce un riporto iniziale e viene composto da EX OR e AND

ADDER A N-BIT

Potrebbero essere composti da half-adder in cascata ma per il ritardo di elaborazione c'è il rischio che si perdano gli ultimi riporti \rightarrow esiste rete che anticipa i riporti. **CARRY-AHEAD**

ARITMETICA BINARIA CON SEGNO

- COMPLEMENTO $\beta-1$ di un numero in base β è un numero A con n cifre, il numero è $(\beta^n - 1) - A$

- Nel caso binario, complemento 1, questo equivale al not di ogni cifra

- COMPLEMENTO A β , dato un numero A di n cifre il numero è $\beta^n - A$

\rightarrow oppure prendi complemento $\beta-1$ e aggiungi 1

\rightarrow oppure $\text{NOT}(A) + 1$

Questo ci serve a implementare la sottrazione come somma e ad introdurre le operazioni con i numeri con segno.

SOTTRACCIONE TRA NUMERI SENZA SEGNO

- Si può sommare il primo numero al complemento a 2 del secondo.

In questo caso il CARRY-OUT A 1 è OK e viceversa

- Viene usato il complemento a 2

$$(A)_2 = a_{n-1} \dots a_0$$

se $A > 0$ normale

se $A < 0$ compl a 2 di $-A$

$$-2^{n-1} \leq A \leq 2^{n-1} - 1$$

- Viene usata perché operazioni tra compl. a 2 \rightarrow compl. a 2 mentre con gli altri modi ciò non è vero

- Per controllare risultati:

$$\text{CARRY-OUT} = \text{CARRY-FLAG}$$

$$\text{FLAG} = 1 \rightarrow \begin{cases} \text{RISULTATO } \neq 0 \text{ e carry} = 1 & \text{OVERFLOW} \\ \text{RISULTATO } \neq 0 \text{ e carry} = 0 & \end{cases}$$

$$\text{FLAG} = 0 \rightarrow \begin{cases} \text{RISULTATO } \neq 0 \text{ e carry} = 0 & \text{OK} \\ \text{RISULTATO } \neq 0 \text{ e carry} = 1 & \end{cases}$$

ALU (Arithmetic Logic Unit)

- Si basano sulla codifica dei numeri senza/con segno
- Sono macchine combinatorie
- Eseguono operazioni aritmetiche e logiche *
- Forniscono risultato e flag \rightarrow segnala particolari condizioni
- Motore delle CPU

* SCELZIONABILI ATTRAVERSO Detti OP-CODE CHE IN BASE AL CARRY-IN IDENTIFICANO L'OPERAZIONE

ARITMETICA \neq LOGICA



STRUTTURA DI
BIT DIPENDENTI



B; F
INDIPENDENTI

↓

RIPORTO



NO RIPORTO

con $M=0$ annulla riporto \rightarrow logica

con $M=1$ non " " \rightarrow aritmetica

↳ c'è come fare half-adder \rightarrow Exor bit a bit

Oltre ai flag di overflow esistono zero flag ($=1$ se ris. è 0) e sign flag ($=1$ se ris. < 0).

09/04/21

RETE SEQUENZIALE

Ho bisogno di dare diverso risultato a una stessa uscita, c'è bisogno di una memoria del passato chiamata STATO

STATO

La rete calcola il suo stato futuro che potrebbe riconfermare o sostituire il suo stato presente.

→ considerando ritardo...

In una rete asincrona ciò avviene istantaneamente, non si tiene conto del tempo

In una rete sincrona ciò può avvenire dopo un Δt .

• Lo stato quindi viene rappresentato in un codice che poi dovrà essere codificato.

y_{k-1}, \dots, y_0 stato presente

y_{k-1}, \dots, y_0 stato futuro

• Avremo K ingressi di stato e N ingressi con M uscite realizzabili con $M+K$ reti combinatorie con $N+K$ ingressi

LA RETE SI SVAGNA SEMPRE IN UNO STATO CASUALE SE NON FACCIO NULLA AL RISUARDO

RETROAZIONE DIRETTA

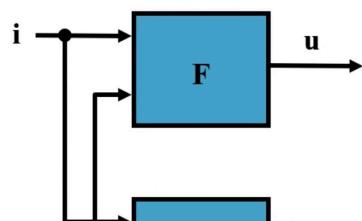
Al venire di un ingresso, la rete ricorda l'ingresso precedente per il tempo che quindi agisce da memoria.

Grazie al ritardo si può mettere la rete in retroazione diretta

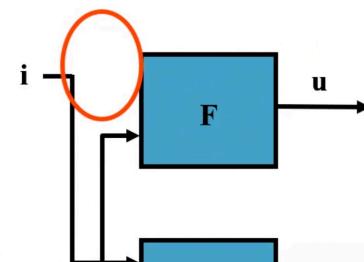
AUTOMA DI MEALY E AUTOMA DI MOORE

Automa di Mealy: quello che abbiamo descritto nella slide precedente

Automa di Moore: caso particolare dell'automa di Mealy, in cui l'uscita dipende solo dallo **stato presente**



Mealy
 $F : S \times I \rightarrow U$



Moore
 $F : S \rightarrow U$

POSSONO SVOLGERE LE STESSSE OPERAZIONI MA CON PRO E CONTRO

GRAFO DEGLI STATI

- NODO → stato presente
 - ARCO → trasizione stato presente - futuro
 - STATO STABILE → arco su se stesso
 - INGRESSI/USCITE → ingressi, uscite
- IN UNA RSA CI DEVE SEMPRE ESSERE UNA STABILITÀ

16/04/21

INDIFFERENZA SULL'USCITA → SOLO SU TRANSIZIONE, MAI SU STABILITÀ

Va usata quando l'uscita cambia valore tra la stabilità nel S.P. e S.F.
Altrimenti ho un GLITCH.

Perché non so quando cambia, per quel poco tempo non importa e decidiamo quel valore nelle mappe. DIMINUIRE VINCOLI SUL PROBLEMA

- NON DEVONO MAI ESSERE GRAFI INTERRUTTIBILI O ASSORBENTI
SOLO FRECCCE USCITE \leftrightarrow SOLO FRECCCE ENTRANTI

DIFERENZA MEENE E MEALY

In MEENE metto l'uscita dentro lo stato (ponerla a fare più stati),
altrimenti si tiene fuori con MEALY

STATO INIZIALE E RESET

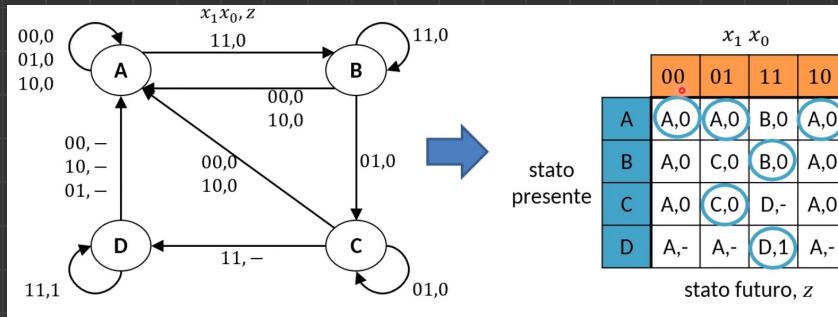
Il reset vale 1 all'inizio e 0 dopo, serve ad entrare la casualità di stato iniziale.

Quindi posso decidere da che stato partire nel grafo, indicandolo con una freccia con RESET

TABELLA DI FLUSSO

"PRESENTI"

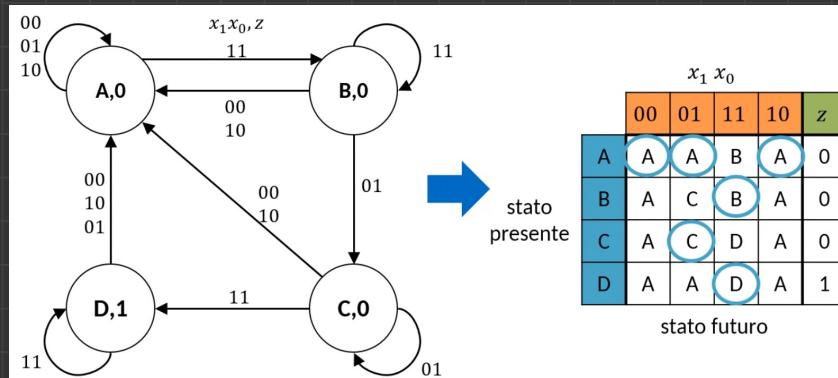
- Ha tante righe quante gli stati e tante colonne quante i bit di ingresso.
- Nelle celle c'è stato futuro e uscita
- Stessa rappr. del grafo ma più ordinata



X MEALY

- Ogni riga deve avere una colonna di stati nulli

-



X MOORE

- IL GRAFO È IL PUNTO DI INIZIO, È PIÙ INTUITIVO
- PER I PASSAGGI DI SINTESI SOBBIAMO USARE LA TABELLA DI FLUSSO

PASSA 6 Gi

GRAFO → TABELLA DI FLUSSO → MODIFICA STATI → TABELLA TRANSIZIONI →
→ MAPPE

COMPORTAMENTO A REGIME E IN TRANSITORIO

- A REGIME = non in transitorio
- IN TRANSITORIO =
 - TRANSITORIO \rightarrow ritardo puro, alea dinamica

ALEA DINAMICA

Quando l'uscita varia più di una volta in transitorio, provocata da diversi ritardi.

Si può eliminare con espressioni PS o SP

ALEA STATICHE

Quanto l'uscita dovrebbe rimanere costante ma in transitorio varia temporaneamente

Per evitarla faccio una sintesi non minima.

19/04/21

REGOLE IMPIEGO CORRETTO RSA

Affinché la rete passi da una cond. di stabilità all'altra in ogni possibile ingresso e S.P. debbo seguire alcune condizioni:

- 1) DURATA INGRESSI
- 2) CODIFICA INGRESSI
- 3) ALEE STATICHE
- 4) CODIFICA DI STATI

DURATA INGRESSI

Esiste un limite superiore alla velocità di funzionamento

- Tempo minimo di durata \rightarrow ritardo massimo tra $T_{Pf0}, \dots, T_{Pfk-1}$ che indichiamo con T_{Pf}
- Bisogna attendere $2T_{Pf}$

CODIFICA DEGLI INGRESSI

È impossibile che due ingressi vengano contemporaneamente, uno viene prima e la rete leggerà quella prima config. portando errori.

- Configurazioni consecutive devono essere adiacenti
- Quindi in alcuni stati ci sono configurazioni impossibili

ALEE STATICHE

- Quando usiamo reti asincrone dobbiamo usare sintesi ridondanti per evitare alee statiche, che porterebbero a transizioni casuali statiche.

CODIFICA DEGLI STATI

- Configurazioni in ingresso a f, che cambia lo stato, devono essere adiacenti

CORSE CRITICHE E NON

- Critica → aleatorietà genera un problema → genera transizione multipla
- Non critica → aleatorietà non genera un problema

Possono esserci entrambi in una tabella

23 / 04 / 21

PREVENZIONE DI CORSE CRITICHE

Sulla tabella di flusso:

- Si possono eliminare a priori le situazioni di corse critiche seguendo le seguenti regole:
 1. Nelle colonne con **una sola stabilità** si inserisce il simbolo dello stato stabile al posto di eventuali condizioni d'indifferenza.
 2. Per le sole colonne **con più stabilità** si traccia il **grafo delle adiacenze**: ad ogni stato è associato un nodo e ad ogni coppia stato presente - stato futuro un ramo orientato che connette i due nodi corrispondenti.
 3. Si sovrappone il grafo ad una mappa con codici di Gray su righe e colonne (come nelle mappe di Karnaugh) per il minimo numero di variabili di stato necessarie e si verifica se è possibile **assegnare configurazioni adiacenti ad ogni coppia di stati coinvolta in una transizione**.
 4. Se è impossibile soddisfare tutti i vincoli di adiacenza, si cerca di ridurli ricorrendo a **transizioni multiple**.
 5. Se non ci si riesce, si **incrementa il numero delle variabili di stato** e si ritorna al punto 3.

ESEMPI SINTESI

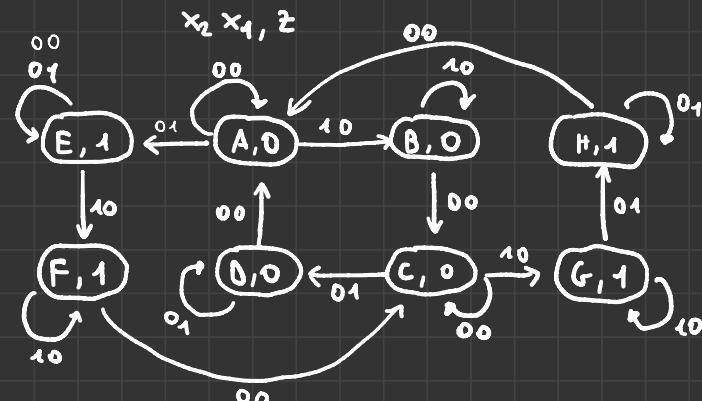
ESEMPIO 1

ingressi: x_1, x_2

uscita: Z

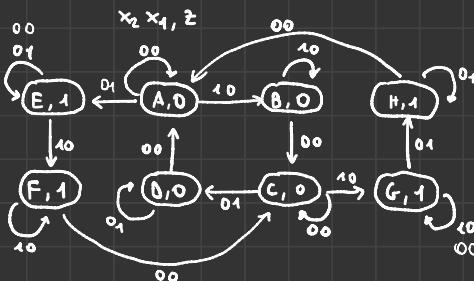
- Grafo:

- A \rightarrow alternanza, 01 - 00
- B \rightarrow alternanza, 10
- C \rightarrow alternanza, 10 - 00
- D \rightarrow alternanza, 01
- E \rightarrow no alternanza, esp. 10
- F \rightarrow alternanza, 10
- G \rightarrow no alternanza, esp. 01
- H \rightarrow alternanza, 01

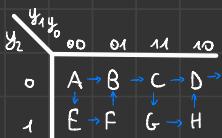


- Tabella di flusso :

	00	01	11	10	Σ
A	(A)	E	-	B	0
B	C	-	-	(B)	0
C	(C)	D	-	G	0
D	A	(D)	-	-	0
E	(E)	(E)	-	F	1
F	C	-	-	(F)	1
G	(G)	H	-	(G)	1
H	A	(H)	-	-	1



- Codifica stati :



	00	01	11	10	Σ
A	(A)	E	-	B	0
B	C	-	-	(B)	0
C	(C)	D	-	G	0
D	A	(D)	-	-	0
E	(E)	(E)	-	F	1
F	B	-	-	(F)	1
G	(G)	H	-	(G)	1
H	D	(H)	-	-	1

	000	001	011	010	100	101	111	110
000	000	100	-	001	0	001	011	0
001	011	-	-	001	0	001	011	0
011	011	010	-	111	0	011	-	0
010	000	010	-	-	0	010	-	0
100	100	100	-	101	1	101	-	1
101	001	-	-	101	1	001	-	1
111	111	110	-	111	1	111	-	1
110	010	110	-	-	1	010	-	1

30/01/21

ANALISI

Percorso inverso a sintesi
Vedi slide

MEMORIE BINARIE

Si basano sulle RSA

Vedremo LATCH SR, LATCH CD, FLIP-FLOP D

LATCH SR

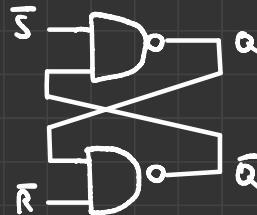
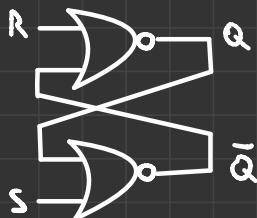
Ingressi Set e Reset \rightarrow \neq del classico reset

Uscite Q e \bar{Q} che rappresenta il bit memorizzato

Comando	S	R	Q
memorizza	0	0	x
scrivi 1	1	0	1
scrivi 0	0	1	0

$S=1, R=1$ è una config. impossibile \rightarrow non vibrerebbe Q e \bar{Q} altimimenti

Si preferisce la sintesi a NOR rispetto a SP perché invece di AND, OR e NOT possono avere solo 2 NOR ed ingressi non negati:



versione a NAND,
Funziona in modo
opposto
↓
LATCH $\bar{S} \bar{R}$

STATO INIZIALE

Si sveglia in stato casuale. \rightarrow priorità rispetto ad S o R
Aggiungendo altri 2 ingressi \rightarrow possiamo inizializzarlo.

RESET (\overline{PRE}) quando è 0 memorizza 1, indip. da S e R
CLEAR (\overline{CLR}) quando è 0 memorizza 0, " " 1 1 1

Uso 2 bit perché devo considerare anche il caso in cui non li uso,
quindi ho 3 config \rightarrow 2 bit

Si chiudono con 0, arrivano bassi.

La rete che gestisce l'inizializzazione è combinatoria.

Se ne può fare la sintesi \rightarrow slide



MALFUNKTIONAMENTO LATCH-SR

1) DURATA DEGLI INGRESSI

Ogni circuito ha bisogno di un certo Δt per memorizzare il bit, se non rispetta questo Δt ma impiega meno tempo rado incontro a un malfunzionamento della RSA che nelle memorie binarie si chiama METASTABILITÀ.

METASTABILITÀ

Il circuito riceve disturbi inevitabilmente, per evitarli bisogna fare sì che i valori dei segnali oscillino sempre o il meno possibile negli intervalli che rappresentano 0 e 1.

Bisogna fare in modo che la rete elabori i segnali in modo tale che i segnali che capitano nella zona di mezzo si spostino il più velocemente possibile e viceversa quelli nelle zone di interesse. Per il primo caso si usa una pendenza > 1 e nel secondo < 1 .

Si manifesta come uscita casuale della memoria e che l'uscita assume il suo valore definitivo più lentamente che se avessi messo ingresso correttamente.

LATCH CD

Comando	C	D	Q
memorizza	0	-	*
scrivi 1	1	1	1
scrivi 0	1	0	0

C - dice scrivi o memorizza
 D - dice cosa scrivere

Non ci sono configurazioni vietate

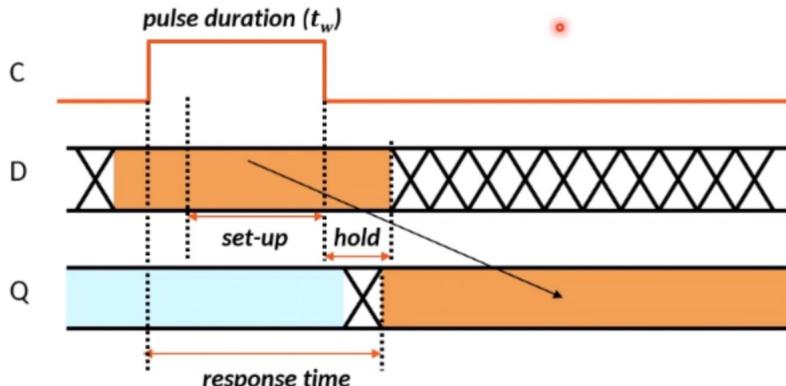
SINTESI DI RETTA

Penso realizzare un LATCH CD partendo da SR.

$$S = CD \quad e \quad R = C\bar{D}$$

I tempi di set-up, di hold e di risposta

- Avendo più gate in cascata, il transitorio del latch CD dura più di quello del latch SR
- Come nel latch SR, anche il comando di campionamento ($C = 1$) deve avere una **durata minima**, indicata ancora come t_w . Il non rispetto di questo vincolo può causare metastabilità come per il latch SR.
- Vi sono altri due tempi da considerare nel latch CD, **rispettivamente prima e dopo il fronte di discesa di C** che chiude un periodo di campionamento:
 - set-up time (t_{su} , tempo di propagazione attraverso i gate)
 - hold time (t_h , tempo necessario per innescare la retroazione)
- D** deve essere **costante** durante $t_{su} + t_h$ per garantire il corretto funzionamento del latch
- Questi vincoli evitano che la rete veda cambiamenti «simultanei» degli ingressi, che come sappiamo sono proibiti in una RSA.
- L'uscita in forma vera e complementata è disponibile a partire dal fronte di salita di C , dopo un tempo detto **response time** (t_r) tipicamente diverso se Q passa da L a H o da H a L



USCITA TRASPARENTE

Se $C=1$ allora $D=Q$ ma con ritardo

Diventa un problema quando cerco di fare dei montaggi in retroazione dell'uscita e dell'ingresso. \rightarrow LOOP INFINITO \rightarrow scrive 0 e 1 ripetutamente

FLIP-FLOP D

Ingresso D per il campionamento e il segnale clock (Δ).

E' progettato per non avere uscite trasparenti.

E' una rete sensibile al fronte:

Positive Edge - triggered \rightarrow fronte salita

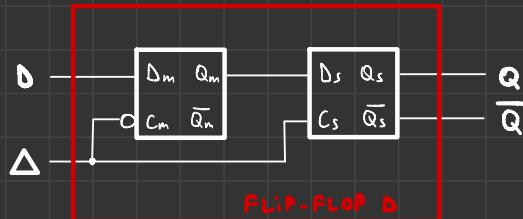
Negative " " " \rightarrow fronte discesa

La rete scrive e memorizza D solo dal momento in cui Δ ha il fronte di salita.

Il valore viene memorizzato fino a fronte di Δ successivo.

FLIP-FLOP D MASTER-SLAVE

Realizzabile con 2 latch CD (MASTER e SLAVE) in cascata.



Ci sono anche CLOCK A DUE FASI, in cui si usano 2 clock per assicurarsi che c'è fase in cui entrambi i latch sono in memorizzazione prima che uno dei due vada in trasparenza.

Non c'è mai un contatto diretto tra D e Q.

non si usano più

I flip flop ora vengono realizzati senza latch, hanno la struttura 2 p. 36.

Hanno 6 NAND, D consuma solo 1 FAN-out, contiene \overline{PNE} e \overline{CLR}

RETI SEQUENZIALI SINCRONE

Introduciamo un dt prima di aggiornare lo stato presente.

↳ COSÌ PRIMA CI ASSICURIAMO CHE SIA TUTTO OK → EVITIAMO MALFUNZIONAMENTI

10/5/21

VANTAGGI

- 1) CODIFICA DEGLI STATI E' ARBITRARIA
- 2) ALLE STATICHE NON SONO PIÙ UN PROBLEMA
- 3) TUTTI GLI INGRESSI POSSONO VARIARE SIMULTANEAEMENTE
- 4) GLI INGRESSI DEVONO RIMANERE COSTANTI FINO A SOVRASCRITTURA S.P.

Usa tanti flip-flop D quanti sono gli stati e mettiamo il periodo T_0 in Δ e in D metto gli S.F. e prendo le uscite come S.P.

MODELLO DI MOORE

CONSEGUenze:

- 1) non c'è mai percorso combinatorio tra ingressi e uscite
↳ USCITA VARIA IN MODO SINCRENO COL CLOCK

2)

:

MODELLO DI MEALY

CONSEGUenze:

- 1) c'è sempre un percorso puramente combinatorio tra ingresso e uscita

:

14/05/24

Sincronizzazione ingressi sincroni

Si usa il Flip-flop D ma c'è rischio metastabilità ma quasi mai.

GRAFO DEGLI STATI

Ogni ingresso s'è

Le frecce non sono più uguali, ogni arco dura un clock e quindi non c'è bisogno di stabilità.

Considero tutte le configurazioni e non metto - per forza

CODIFICA

Nessun vincolo

SINTESI DIRETTA

REGISTRO, SHIFT-REGISTER, CONTATORE

REGISTRO a K bit

Rete logica sincrona

Memorizza K bit

INGRESSI:

- WE → write enable se 1 scrive, se 0 ricorda
- IN → K ingressi
- A RESET → reset asincrono

Usiamo un Flip-flop D con un mux che ripende l'uscita in 0 e

IN in 1 e usa WE per scegliere

se A RESET

Usiamo un Flip-flop D

SHIFT - REGISTER

Memorizza gli ultimi K bit ricevuti da IN e li rende disponibili con OUT

UNIVERSAL SHIFT - REGISTER

Comando	S_1	S_0	$(OUT\ i)^{n+1}$
Hold	0	0	$(OUT\ i)^n$
Shift right	0	1	$(OUT\ i - 1)^n$
Shift left	1	0	$(OUT\ i + 1)^n$
Load	1	1	$(I\ i)^n$

17/05/21

CONVERSIONE DATI

- $S \rightarrow P$:
 - MOORE =
 - MEALY =

- $P \rightarrow S$:

serve comando di LOAD

MOLTIPLICAZIONE / DIVISIONE

SHIFT ARITMETICO

RICORDATORI SEQUENZIALI

MONO IMPULSORE

Asseverisce la sua uscita esattamente per un ciclo di clock quando l'ingresso, anche asincrono, ha un fronte di salita.

21/03/21

CONTATTORE

- MODULO N → conta primi N numeri binari
- Lo posso realizzare con un Adder attaccato a 2 FB-D
 - MOORE
 - POCO EFFICACE → soluzione p. 86
- Per continuare a contare devo tenere ENABLE = 1

es.

